**Računalniška arhitektura**

Pri **prvi nalogi** sem najprej v r0 naložil lokacijo NIZ1 in prištel 4 zaradi tega, ker moramo začeti z črko s, torej spustimo “Dane”. In v r4 sem naložil lokacijo NIZ2. Nato sem naredil zanko, v kateri se v r1 naloži vrednost lokacije registra r1, torej prvo črko. Primerjal sem vrednost r1 z konstanto #96, ki je v ascii na mestu med velikimi in malimi črkami. In potem sem, če je r1 večji od 96, potem je v r2 zapisal razliko med r1 in 32. 32 pa zato, ker je to razmak med veliko in malo črko v ascii primer med s in S. In seveda, potem preveri, če je r1 manjši od 96 pa registru v r2 zapiše r1 plus 32. Nato imam ponovno stavek cmp, ki mi preveri, če je r2 #0x40, kar pomeni, da je bila prejšnja vrednost 0x20 to pa je znak za presledek v ascii. Torej presledku ne želim prišteti nobene vrednosti ampak sem je že prej torej jo sedaj želim odšteti in sem v naslednjem koraku v r2 zapisal r2 minus #32 ali drugače namesto #32 bi lahko tudi #0x20 kar je isto, le v drugem zapisu. Ker sem dobil željeno vrednost, sem sedaj iz registra r2, kjer je moja spremenjena vrednost zapisal na lokacijo r4, kjer se začne moj NIZ2 prav r2. Nato sem še prištel registru r0 vrednost 1, da se premakne do drugega znaka NIZ1. In r4 sem prištel tudi 1 zato, da sem se tudi v NIZ2 premaknil, da bom lahko v novem loop zapisal na naslednjo lokacijo. Nato samo na koncu še preverim kdaj r0 doseže lokacijo 0x30, ki je konec NIZ1, kar pomeni za zaključim z zanko loop.

Pri **drugi nalogi** sem najprej dodal spremenljivki MIN in MIN\_INDEKS. Nato sem od vseh treh spremenljivk zapisal lokacije v r0(tabela), r1(min) in r2(min\_indeks). V register r5 in register r6 sem še zapisal konstanto 0. Ena bo služila kot števec in sicer r5, ki se bo vsakič ko vstopimo v nov loop povečala za ena, druga pa bo le držala vrednost indeksa najmanjšega števila. Nato vstopimo v zanko loop, ki v register r3 naloži vrednost lokacije registra r0 in v r4 naloži vrednost lokacije r1. A naloži v ukazom ldrsh, ki pomeni load in s stoji za tem, da nam zapisuje zastavice, ki jih potrebujemo za primerjanje in h stoji za hword. Potem primerjamo r3 in r4, če je r3 manjši od r4, kar pomeni da je trenutna vrednost manjša od minimuma, potem se v r4 v minimum zapiše r3 trenutna vrednost. Nato iz r4 naložimo vrednost v pomnilnik, ki je na lokaciji r1 in iz registra r6 naložimo v pomnilnik vrednost z lokacije registra r2. V naslednji fazi r0 dodamo vrednost 2, ker imamo vrednost velike 16b torej potrebujemo premik za #2. In registru r5 prištejemo 1, kar pomeni da smo števec povečali za ena. Na koncu ponovno preverimo ali je r0 dosegel lokacijo 0x36, kar je konec naše tabele. In ko je r0 enak 0x36 smo izstopili iz zanke in naš program se konča.